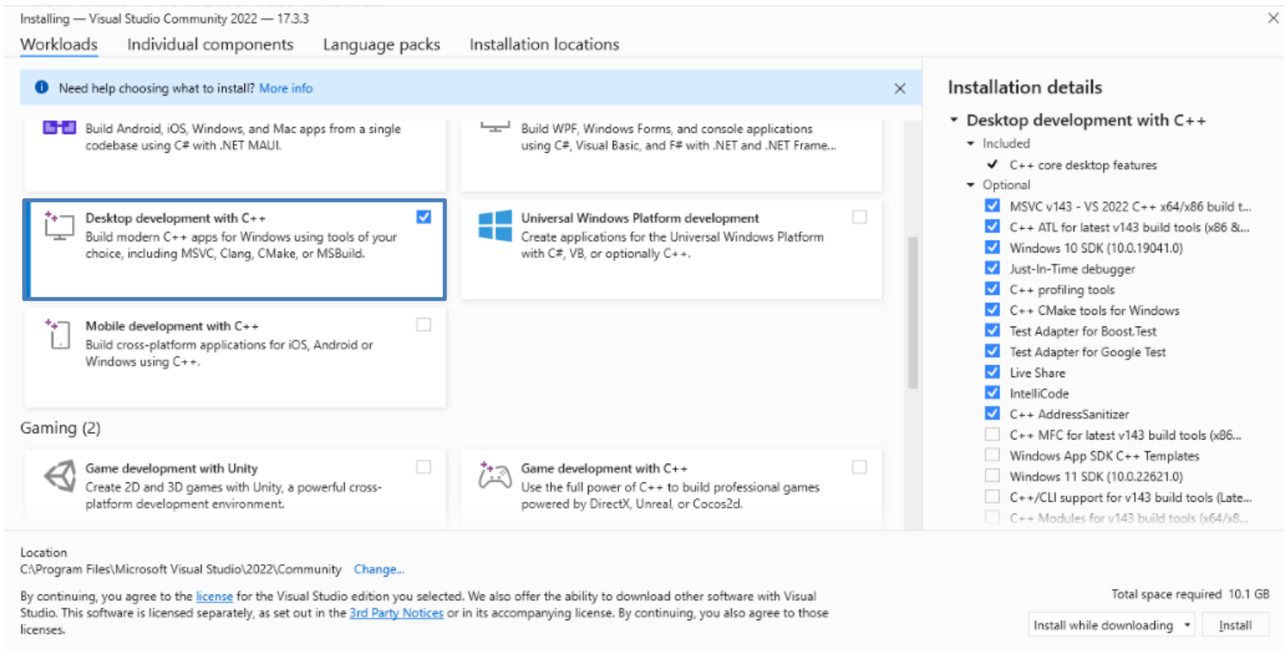
OpenMPI Installation and Setup on Windows 10

1. Download and install Microsoft Visual Studio Community 2022 (or newest version). Choose the community version as it is free and good enough for what we need.

2. Select the 'Desktop development with C++' workload, and install. It can take several minutes for all the packages to download.



Installing — Visual Studio Community 2022 — 17.3.3

Workloads | Individual components | Language packs | Installation locations

Need help choosing what to install? More info

Build Android, iOS, Windows, and Mac apps from a single codebase using C# with .NET MAUI.

Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET and .NET Frame...

☑ **Desktop development with C++** — Build modern C++ apps for Windows using tools of your choice, including MSVC, Clang, CMake, or MSBuild.

☐ **Universal Windows Platform development** — Create applications for the Universal Windows Platform with C#, VB, or optionally C++.

☐ **Mobile development with C++** — Build cross-platform applications for iOS, Android or Windows using C++.

Gaming (2)

☐ **Game development with Unity** — Create 2D and 3D games with Unity, a powerful cross-platform development environment.

☐ **Game development with C++** — Use the full power of C++ to build professional games powered by DirectX, Unreal, or Cocos2d.

Installation details

- Desktop development with C++
  - Included
    - ✓ C++ core desktop features
  - Optional
    - ☑ MSVC v143 - VS 2022 C++ x64/x86 build t...
    - ☑ C++ ATL for latest v143 build tools (x86 &...
    - ☑ Windows 10 SDK (10.0.19041.0)
    - ☑ Just-In-Time debugger
    - ☑ C++ profiling tools
    - ☑ C++ CMake tools for Windows
    - ☑ Test Adapter for Boost.Test
    - ☑ Test Adapter for Google Test
    - ☑ Live Share
    - ☑ IntelliCode
    - ☑ C++ AddressSanitizer
    - ☐ C++ MFC for latest v143 build tools (x86...
    - ☐ Windows App SDK C++ Templates
    - ☐ Windows 11 SDK (10.0.22621.0)
    - ☐ C++/CLI support for v143 build tools (Late...
    - ☐ C++ Modules for v143 build tools (x64/x8...

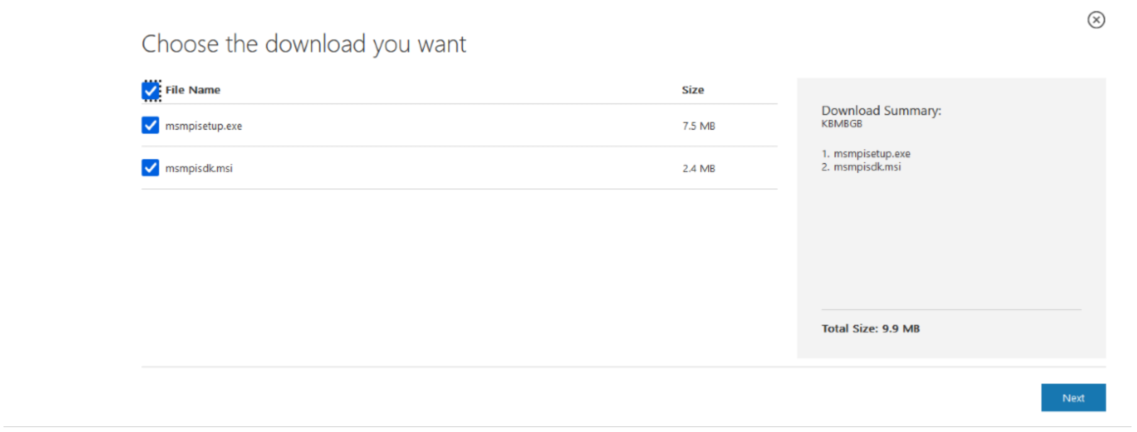
Location
C:\Program Files\Microsoft Visual Studio\2022\Community Change...

By continuing, you agree to the license for the Visual Studio edition you selected. We also offer the ability to download other software with Visual Studio. This software is licensed separately, as set out in the 3rd Party Notices or in its accompanying license. By continuing, you also agree to those licenses.

Total space required 10.1 GB

Install while downloading | Install

3. Download MS-MPI SDK and Redist installers. Select both **msmpisetup.exe** and **msmpisdk.msi** and install them.



Choose the download you want

☑ File Name	Size
☑ msmpisetup.exe	7.5 MB
☑ msmpisdk.msi	2.4 MB

Download Summary:
KBMBGB

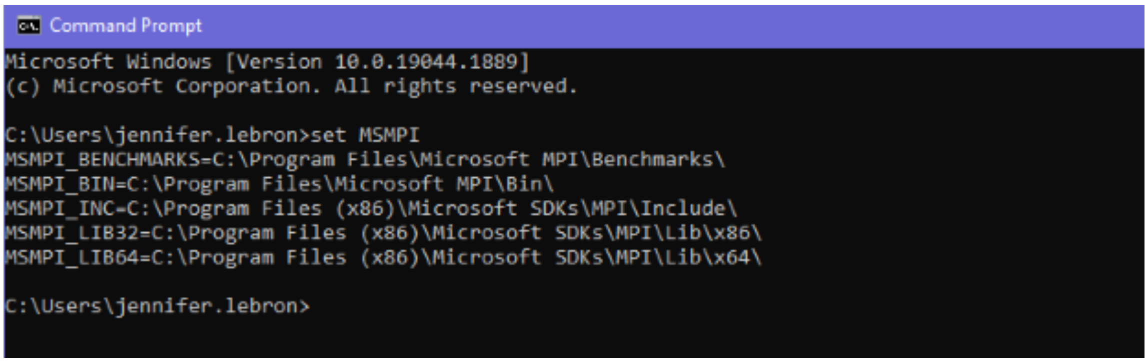
1. msmpisetup.exe
2. msmpisdk.msi

Total Size: 9.9 MB

Next

4. After installation, you can verify that the MS-MPI environment variables have been set correctly (you will want to use these env vars in Visual Studio).

To do so, open Command Prompt and type '`set MSMPI`'. This should produce output similar to the below:



```
Command Prompt
Microsoft Windows [Version 10.0.19044.1889]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jennifer.lebron>set MSMPI
MSMPI_BENCHMARKS=C:\Program Files\Microsoft MPI\Benchmarks\
MSMPI_BIN=C:\Program Files\Microsoft MPI\Bin\
MSMPI_INC=C:\Program Files (x86)\Microsoft SDKs\MPI\Include\
MSMPI_LIB32=C:\Program Files (x86)\Microsoft SDKs\MPI\Lib\x86\
MSMPI_LIB64=C:\Program Files (x86)\Microsoft SDKs\MPI\Lib\x64\

C:\Users\jennifer.lebron>
```

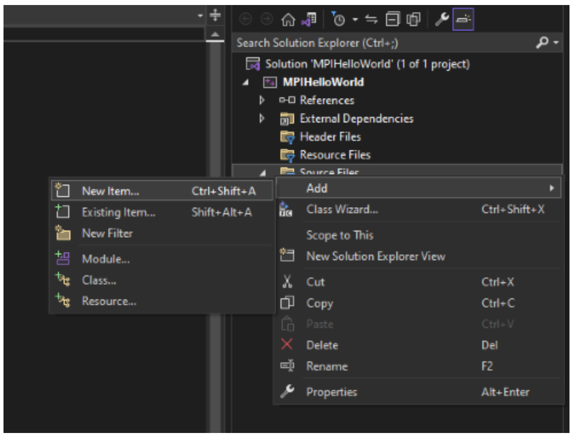
If you get errors, set the Environment and restart your machine and execute it again.

5. Open Visual Studio 2022 and create a **new Project > Console App**. Let's name our project `MPIHelloWorld` and use default settings. This should create by default a .cpp file within the Source Files folder named the same as the project.

If you do not have the 'Desktop development with C++' workload in Visual Studio:

Scroll to the bottom of project templates and click 'Install more tools and features'. This will take you to the Visual Studio Installer and should show your current installation. Click 'modify' and this should bring up the workloads window. At this point you can select 'Desktop development with C++' and install.

If you are unable to install this workload, follow the process below:



Create a new Empty Project (Console App), name it MPIHelloWorld with default settings.

Click on the project name (i.e. MPIHelloWorld) in the Solution Explorer and add a new C++ Source File (Project > Add > New Item > C++ File) and name it MPIHelloWorld.cpp

---

6. Now copy the **'hello world' code** below into your MPIHelloWorld.cpp class.

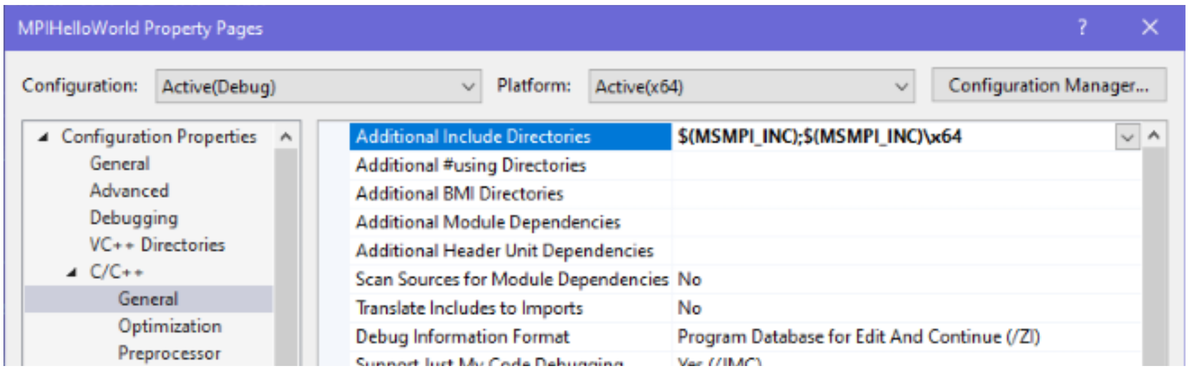
```cpp
#include <iostream>
#include <mpi.h>
int world_size, world_rank;
int main(int argc, char** argv) {
    MPI_Init(NULL, NULL);
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
    std::cout << "Hello " << world_rank << ", World: " << world_size << std::endl;
    // finalise MPI and return control to the OS;
    MPI_Finalize();
    return 0;
}
```

It will show some errors as you've not yet configured this particular project for the MPI Library. Let's do that first before we compile it.

7. Open the "Property Pages" (Project -> Properties). Add the following in the **Additional Include Directories** area and press 'Apply'.

If building for x64: **`$(MSMPI_INC);$(MSMPI_INC)\x64`**

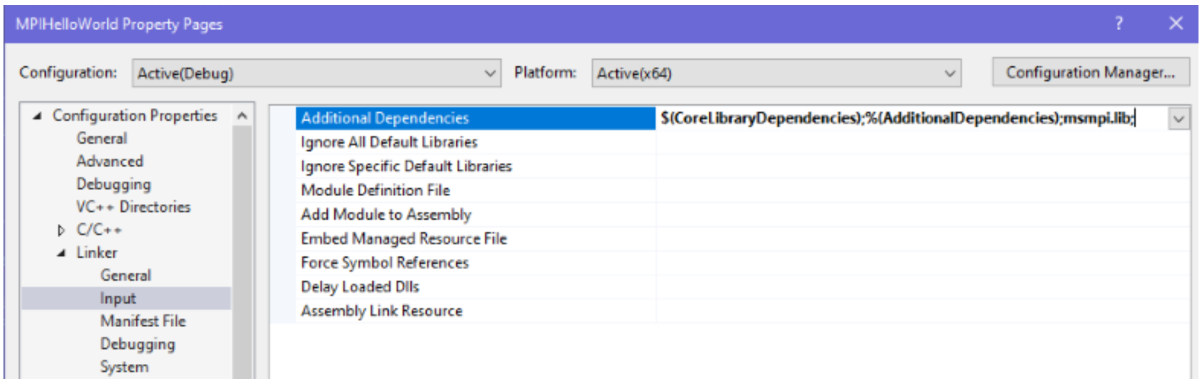
If building for x86: **`$(MSMPI_INC);$(MSMPI_INC)\x86`**



Configuration: Active(Debug) Platform: Active(x64)

Additional Include Directories	$(MSMPI_INC);$(MSMPI_INC)\x64
Additional #using Directories	
Additional BMI Directories	
Additional Module Dependencies	
Additional Header Unit Dependencies	
Scan Sources for Module Dependencies	No
Translate Includes to Imports	No
Debug Information Format	Program Database for Edit And Continue (/ZI)
Support Just My Code Debugging	Yes (/JMC)

8. Setup the linker library by adding `msmpi.lib;` to the **Additional Dependencies**



Configuration: Active(Debug) Platform: Active(x64)

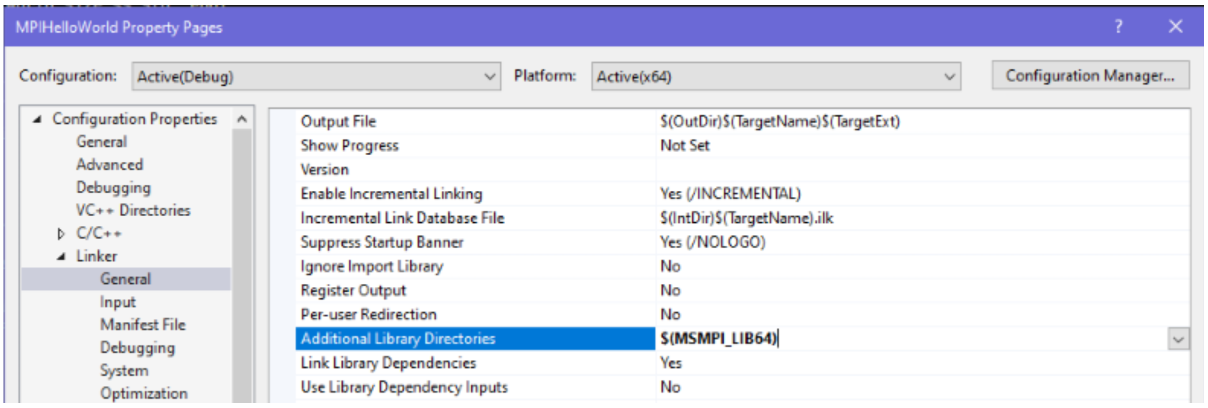
Additional Dependencies	$(CoreLibraryDependencies);%(AdditionalDependencies);msmpi.lib;
Ignore All Default Libraries	
Ignore Specific Default Libraries	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded Dlls	
Assembly Link Resource	

Don't forget the semicolon (;) after the msmpi.lib to separate other dependencies from the newly included dependency.

9. Add the following to the **Additional Library Directories**.

If building for x64: `$(MSMPI_LIB64)`

If building for x86: `$(MSMPI_LIB32)`



Output File	$(OutDir)$(TargetName)$(TargetExt)
Show Progress	Not Set
Version	
Enable Incremental Linking	Yes (/INCREMENTAL)
Incremental Link Database File	$(IntDir)$(TargetName).ilk
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	$(MSMPI_LIB64)
Link Library Dependencies	Yes
Use Library Dependency Inputs	No

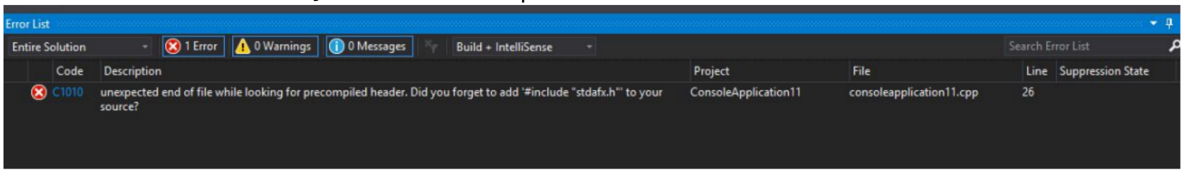
10. Now, Build the solution by selecting Build > Build Solution (or press `Ctrl+Shift+B`). This will compile your code and if successful, creates an executable file in the Debug folder of your project.

You should see a message like this:

```
Build started...
1>------ Build started: Project: MPIHelloWorld, Configuration: Debug x64 ------
1>MPIHelloWorld.cpp
1>MPIHelloWorld.vcxproj -> C:
\Users\jennifer.lebron\source\repos\MPIHelloWorld\x64\Debug\MPIHelloWorld.exe
```

* If you get any linking error messages, it is most likely you're building for 64 bits yet specifying 32 bits linking libraries.

** Get this error after you build/compile? (C1010)



Code	Description	Project	File	Line	Suppression State
C1010	unexpected end of file while looking for precompiled header. Did you forget to add '#include "stdafx.h"' to your source?	ConsoleApplication11	consoleapplication11.cpp	26	

Go to Property Pages – C/C++ Precompiled Header then click Precompiled Header and select **Not Using Precompiled Headers**. *Note: Precompiled headers can be turned off when you create a new project!*

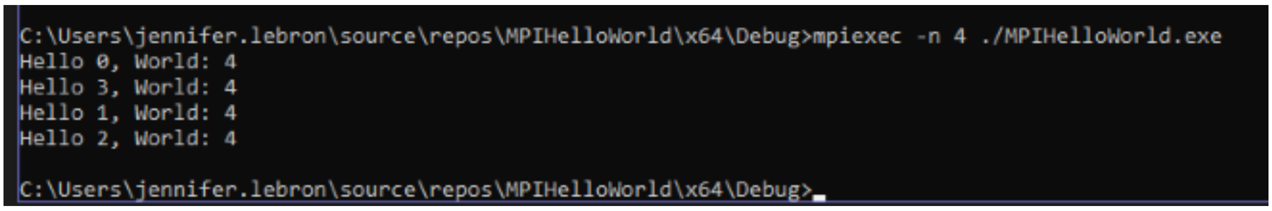
11. Lastly, lets run the program. Open Command Prompt and navigate to the **Debug** directory. The path to this directory is output in the build message (see above).

    Execute your code following this command template:

    ```
    mpiexec -n <number of processing elements> ./<nameofExecutableFile>
    ```

    Expected output when running MPIHelloWorld using:

    `mpiexec -n 4 ./MPIHelloWorld.exe`

    ```
    C:\Users\jennifer.lebron\source\repos\MPIHelloWorld\x64\Debug>mpiexec -n 4 ./MPIHelloWorld.exe
    Hello 0, World: 4
    Hello 3, World: 4
    Hello 1, World: 4
    Hello 2, World: 4

    C:\Users\jennifer.lebron\source\repos\MPIHelloWorld\x64\Debug>_
    ```

    When using the MPI library, the executable given in the mpiexec command will run on the amount of nodes specified by the `-n` flag. The code then runs concurrently on `n` amount of nodes, and each node will be assigned a ‘rank’ from 0 to n-1.

Keep in mind you will need to follow Steps 7-11 for each new Project you create in Visual Studio in order to use the MPI Library.